Applied Geometric Computing and CGAL

Number Types

Ron Wein

wein@tau.ac.il

Tel Aviv University

Overview

- Number types for certified computations,
- Separation bounds,
- Numeric filters.

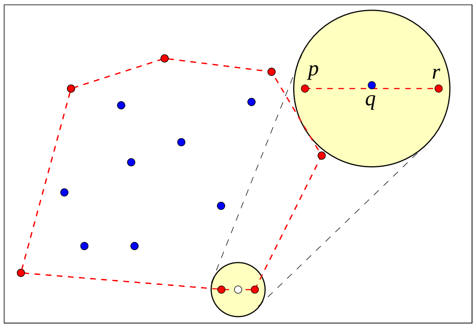
Machine-Precision Reals

The representation of floating-point numbers (the IEE 754 standard):

$$\begin{aligned} \texttt{float} &: (-1)^S \times 1.M \times 2^{E-127} \\ \texttt{double} &: (-1)^S \times 1.M \times 2^{E-1023} \end{aligned}$$

Convex-Hull Computation

Is q part of the convex hull?



Zooming in ... No!

Unbounded Integers

Suppose that all points have integer coordinates.

Given three points p, q and r, we need to evaluate the sign of the expression:

$$\vec{pq} \times \vec{pr} = \det \begin{pmatrix} p_x & p_y & 1 \\ q_x & q_y & 1 \\ r_x & r_y & 1 \end{pmatrix}$$

Danger of integer overflow (numbers larger than 2^{31}) ...

Need to use unbounded integers.

Unbounded Integer Types

- Constructed from `int`, `long` or a string.
- Have exact $\{+, -, \times\}$ operators (and exact comparisons).
- Supported by several libraries:
  - GMP (Gnu multi-precision) : `gmpz` (wrapped by `CGAL::Gmpz`),
  - LEDA (MPII Saarbrücken) : `integer`,
  - CORE (NYU) : `BigInt` (based on `gmpz`),
  - User defined classes.

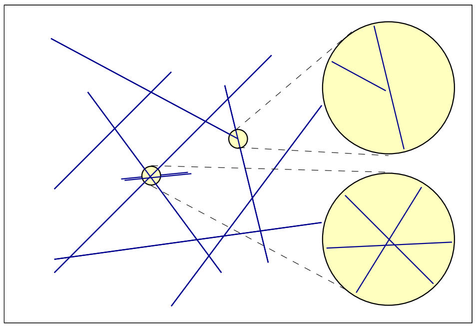
Big Floating-Point Types

Alternative approach — use floating-point numbers with unbounded mantissa (and perhaps unbounded exponent).

- Constructed from `int`, `long` or from `float`, `double`.
- Have exact $\{+, -, \times\}$ operators (and exact comparisons).
- Supported by several libraries:
  - CGAL : `MP_Float` (bounded exponent),
  - LEDA : `bigfloat`,
  - CORE : `BigFloat`.

Constructing an Arrangement of Segments

Determine the exact topology ...



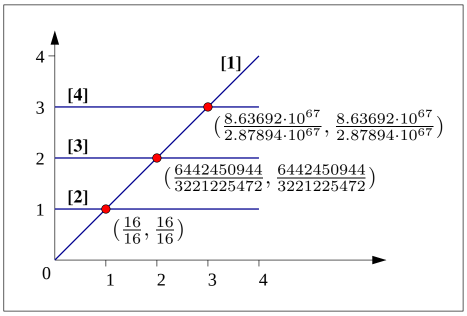
Zooming in ... Tricky!

Exact Rational Types

- Constructed from pairs of (unbounded) integers.
- Have exact $\{+, -, \times, \div\}$ operators (and exact comparisons).
- Supported by several libraries:
  - CGAL : `Quotient<NT>`,
  - GMP : `gmpq` (wrapped by `CGAL::Gmpq`),
  - LEDA : `rational`,
  - CORE : `BigRat` (based on `gmpq`),
  - User defined classes.

The number-types supplied by GMP and LEDA can be canonicalized (automatically or upon request).

The Danger of Working with Rationals



- Use normalizations, based on GCD computation.
- Indiscriminate normalization $\longrightarrow$ running time penalty.

Key Observation

We usually do not care what is the exact value of an expression, we just need to know its sign:

- In predicates such as orientation or in-circle.
- Comparing A and B by evaluating the sign of $A - B$.

This observation leads to the idea of floating-point filters:

- Static [Karasick, Lieber, Nackman 1991] [Fortune, van Wyk 1993],
- Semi-dynamic [Schewchuk 1996],
- Dynamic [Mehlhorn, Näher 1994].

Floating-Point Filters

In order to determine the sign of a rational expression E, compute a floating-point approximation $\tilde{E}$ of E, with a guaranteed approximation error δ:

$$|E - \tilde{E}| < \delta$$

If $|\tilde{E}| > \delta$, then $\mathrm{sign}(E) = \mathrm{sign}(\tilde{E})$.

Otherwise, resort to exact computations with rationals.

Floating-Point Filters (II)

Theorem [Funke 1997] The approximation error of an expression E using floating-point number with p bits in their mantissa is bounded by:

$$\delta = 2^{-p} \odot E_{\text{ind}} \odot \widetilde{E_{\text{sup}}}\ ,$$

where $\widetilde{E_{\text{sup}}}$ and E_{ind} are recursively defined:

E	$\widetilde{E}$	$\widetilde{E_{\text{sup}}}$	E_{ind}
A	A	$\lvert A\rvert$	0
$A+B$	$\widetilde{A}\oplus\widetilde{B}$	$\widetilde{A_{\text{sup}}}\oplus\widetilde{B_{\text{sup}}}$	$1+\max(A_{\text{ind}}, B_{\text{ind}})$
$A-B$	$\widetilde{A}\ominus\widetilde{B}$	$\widetilde{A_{\text{sup}}}\oplus\widetilde{B_{\text{sup}}}$	$1+\max(A_{\text{ind}}, B_{\text{ind}})$
$A\times B$	$\widetilde{A}\odot\widetilde{B}$	$\widetilde{A_{\text{sup}}}\odot\widetilde{B_{\text{sup}}}$	$1+A_{\text{ind}}+B_{\text{ind}}$
$A\div B$	$\widetilde{A}\oslash\widetilde{B}$	$\frac{(\lvert\widetilde{A}\rvert\oslash\lvert\widetilde{B}\rvert)\oplus(\widetilde{A_{\text{sup}}}\oslash\widetilde{B_{\text{sup}}})}{(\lvert\widetilde{B}\rvert\oslash\lvert\widetilde{B_{\text{sup}}}\rvert)\ominus(B_{\text{ind}}+1)\cdot 2^{-p}}$	$1+\max(A_{\text{ind}}, B_{\text{ind}}+1)$

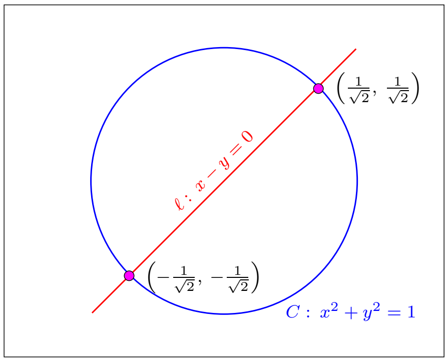
Interval Arithmetic

[Brönnimann, Burnikel, Pion 1998] Maintain an interval $[E_{\text{low}}, E_{\text{high}}]$ of floating-point number that contain the true value of E.

The sign of E can be easily determined if $0 \notin [E_{\text{low}}, E_{\text{high}}]$.

Widely used in CGAL for filtering predicates.

Irrational Numbers



$\sqrt{2}$ is not rational, as discovered by Hippasus of Metapontum (ca. 500 BC).

Algebraic Numbers

Definition A number $\alpha \in \mathbb{R}$ is called algebraic if there exist integer coefficients $c_0, c_1, \dots c_d \in \mathbb{Z}$ such that α is a root of the polynomial:

$$P(x) = c_d \cdot x^d + \dots + c_1 \cdot x + c_0$$

If P divides any other polynomial with integer coefficients $Q \in \mathbb{Z}[x]$ such that $Q(\alpha) = 0$, we say that α is an algebraic number of degree d.

Observation Every rational number $q \in \mathbb{Q}$ is an algebraic number of degree 1.

Algebraic Numbers (II)

- The set $\mathbb{A}$ of algebraic numbers is closed under the $\{+, -, \times, \div, \sqrt[k]{\cdot}\}$ operators:

  $\sqrt{2}\,, \quad \sqrt{3} + \sqrt{5}\,, \quad \frac{2+\sqrt[3]{7}}{\sqrt{61}-4}\,.$

- In addition, if $\alpha_0, \alpha_1, \dots, \alpha_d \in \mathbb{A}$ then all roots of the polynomial $\sum_{i=0}^{d} \alpha_i \cdot x^i$ are also algebraic.

Are all real numbers algebraic? ($\mathbb{A} \stackrel{?}{=} \mathbb{R}$)
No! π and e, for example are transcendental (not algebraic).

- In fact $|\mathbb{A}| = \aleph_0$ (while $|\mathbb{R}| = \aleph_1$),
  So most real numbers are not algebraic.

Sylvester Matrices

Definition Let $F(x) = \sum_{i=0}^{m} a_i x^i$ and $G(x) = \sum_{i=0}^{n} b_i x^i$ be two polynomials. The Sylvester Matrix of F and G is the $(m+n) \times (m+n)$ matrix:

$$\mathcal{M}(F,G) = \begin{pmatrix} a_m & a_{m-1} & & \cdots & a_0 & 0 & \cdots & 0 \\ 0 & a_m & a_{m-1} & & \cdots & a_0 & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & & & \ddots & \ddots \\ 0 & \cdots & 0 & a_m & \cdots & & \cdots & a_0 \\ b_n & b_{n-1} & & \cdots & b_0 & 0 & \cdots & 0 \\ 0 & b_n & b_{n-1} & & \cdots & b_0 & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & & & \ddots & \ddots \\ 0 & \cdots & 0 & b_n & \cdots & & \cdots & b_0 \end{pmatrix} \begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} n \text{ rows} \\ \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} m \text{ rows} \end{array}$$

Resultants

Definition The resultant of the two polynomials $F(x)$ and $G(x)$ is defined to be the determinant of their Sylvester matrix:

$$\mathrm{res}(F, G) = \det(\mathcal{M}(F, G))$$

Theorem If the roots of the two polynomials are $\alpha_1, \ldots, \alpha_m$ and $\beta_1, \ldots, \beta_n$ (such that $F(x) = f_m \prod_{i=1}^{m} (x - \alpha_i)$ and $G(x) = g_n \prod_{j=1}^{n} (x - \beta_j)$), then:

$$\mathrm{res}(F, G) = \prod_{i=1}^{m} \prod_{j=1}^{n} (\alpha_i - \beta_j)$$

Usage of Resultants

Theorem Given two polynomials $A(x) = a_m \prod_{i=1}^{m} (x - \alpha_i)$ and $B(x) = b_n \prod_{j=1}^{n} (x - \beta_j)$ in $\mathbb{Z}[x]$.
Then we can construct a polynomial $R(x) \in \mathbb{Z}[x]$ of degree mn, whose roots are γ_{ij}:

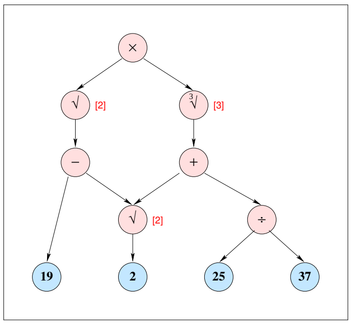
$$\begin{array}{lcl}
R(x) = \mathrm{res}(A(x - y), B(y)) & \Longrightarrow & \gamma_{ij} = \alpha_i + \beta_j \\
R(x) = \mathrm{res}(A(x + y), B(y)) & \Longrightarrow & \gamma_{ij} = \alpha_i - \beta_j \\
R(x) = \mathrm{res}(y^m A(\frac{x}{y}), B(y)) & \Longrightarrow & \gamma_{ij} = \alpha_i \cdot \beta_j \\
R(x) = \mathrm{res}(A(xy), B(y)) & \Longrightarrow & \gamma_{ij} = \frac{\alpha_i}{\beta_j}
\end{array}$$

Algebraic Number Types (basic)

- Constructed from unbounded integers, rationals or floating-point numbers.
- Have exact $\{+, -, \times, \div, \sqrt{\cdot}\}$ operators (and exact comparisons).
- Algebraic number libraries:
  - LEDA : `real`
    (supports also the $\sqrt[k]{\cdot}$ operator),
  - CORE : `Expr` .

The Expression DAG

Algebraic numbers are represented using a directed acyclic graph that encodes the underlying expression.



$$\sqrt{19-\sqrt{2}} \cdot \sqrt[3]{\sqrt{2}+\frac{25}{37}}$$

Separation Bounds

Theorem [Mignotte 1980] For each algebraic expression E there exists a lower bound $\sigma(E)$ on the absolute value of E, such that either $|E| > \sigma(E)$ or $E = 0$.

1. Let $i \longleftarrow 0$ and $\delta_0 \longleftarrow \delta$ (caused by floating-point approximation).
2. Compute an approximation $\tilde{E}_i$ such that $|E - \tilde{E}_i| < \delta_i$.
3. If $|\tilde{E}_i| \geq \delta_i$, then $E \neq 0$, and the sign of E is known — stop.
4. If $|\tilde{E}_i| < \delta_i \leq \frac{\sigma(E)}{2}$, then $E = 0$ — stop.
5. Increment i and let $\delta_{i+1} \longleftarrow \frac{\delta_i}{2}$. Goto step 2.

Separation Bounds (II)

Lemma Let α be an algebraic number of degree d and let U be an upper bound on the absolute value of all conjugates of α. The following holds:

$$|\alpha| \geq U^{1-d}$$

Proof Consider $\sum_{i=0}^{d} c_i \cdot x^i$, the minimal polynomial defining α:

$$1 \leq |c_0| = \left|\prod_{k=1}^{d} \alpha_k\right| = |\alpha| \cdot \prod_{k=2}^{d} |\alpha_k| \leq |\alpha| \cdot U^{d-1}$$

The BFMS Bound

Theorem [Burnikel, Fleischer, Mehlhorn, Schirra 2000]
The separation bound for an expression E can be easily evaluated as:

$$|\mathrm{val}(E)| \geq \left(l(E) u(E)^{\deg(E)-1} \right)^{-1} ,$$

where $u(E)$ and $l(E)$ are recursively defined:

E	$u(E)$	$l(E)$
$n \in \mathbb{Z}$	$\|n\|$	1
$E_1 \pm E_2$	$u(E_1)l(E_2) + l(E_1)u(E_2)$	$l(E_1)l(E_2)$
$E_1 E_2$	$u(E_1)u(E_2)$	$l(E_1)l(E_2)$
E_1 / E_2	$u(E_1)l(E_2)$	$l(E_1)u(E_2)$
$\sqrt[k]{E_1}$ and $u(E_1) \geq l(E_1)$	$\sqrt[k]{u(E_1)l(E_1)^{k-1}}$	$l(E_1)$
$\sqrt[k]{E_1}$ and $u(E_1) < l(E_1)$	$u(E_1)$	$\sqrt[k]{u(E_1)^{k-1}l(E_1)}$

The BFMS Bound (proof)

We always define the expression to be division-free, that is $E = \frac{E_u}{E_l}$. We do so by the following transformations:

$$\frac{A_u}{A_l} \pm \frac{B_u}{B_l} = \frac{A_u B_l \pm A_l B_u}{A_l B_l} \quad , \quad \frac{A_u}{A_l} \cdot \frac{B_u}{B_l} = \frac{A_u B_u}{A_l B_l} \quad , \quad \frac{A_u}{A_l} / \frac{B_u}{B_l} = \frac{A_u B_l}{A_l B_u}$$

$$\sqrt[k]{\frac{A_u}{A_l}} = \frac{\sqrt[k]{A_u}}{\sqrt[k]{A_l}} = \frac{\sqrt[k]{A_u A_l^{k-1}}}{A_l} = \frac{A_u}{\sqrt[k]{A_u^{k-1} A_l}}$$

Assume $E = \frac{\beta}{\gamma}$ and let $u(E), l(E)$ be the upper bounds on the conjugates of β, γ. If $E \neq 0$, then $\beta \neq 0$ and we have:

$$|E| = \frac{|\beta|}{|\gamma|} \geq \frac{1}{u(E)^{\deg(\beta)-1}} \cdot \frac{1}{l(E)} \geq \frac{1}{u(E)^{\deg(E)-1} \cdot l(E)}$$

Performance of the Floating-Point Filter

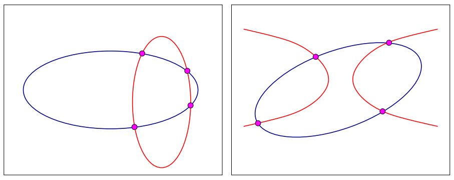
x	Comparison	double Corr.	double Time	leda_real Corr.	leda_real Time	CORE::Expr Corr.	CORE::Expr Time
$-\frac{b}{a}$	$ax + b = 0$	2.6%	0.03	100%	95	100%	184
	$ax + b = 1$	100%	0.03	100%	1.5	100%	2
$\frac{b \pm \sqrt{b^2-4ac}}{2a}$	$ax^2 + bx + c = 0$	0.81%	0.05	100%	224	100%	814
	$ax^2 + bx + c = 1$	100%	0.05	100%	2.5	100%	2.5
$\sqrt{\frac{b \pm \sqrt{b^2-4ac}}{2a}}$	$ax^4 + bx^2 + c = 0$	0.57%	0.07	100%	1133	100%	1764
	$ax^4 + bx^2 + c = 1$	100%	0.07	100%	3.5	100%	3.5

- Results averaged over 100,000 comparisons.
- $a, b, c \in \mathbb{N}$ randomly chosen from $[-1000, 1000]$.
- Times are given in microseconds.
- Using LEDA 4.0 and CORE 1.5 (faster version are now available).

Working with Conics

Consider curves of the form ($r, s, t, u, v, w \in \mathbb{Z}$):

$$r \cdot x^2 + s \cdot y^2 + t \cdot xy + u \cdot x + v \cdot y + w = 0$$



Coordinates of the intersection point are algebraic numbers of degree 4.

Working with Algebraic Curves

Given two curves $A(x,y) = \sum_{i=0}^{d} \sum_{j=0}^{d-i} a_{ij} x^i y^j$ and $B(x,y) = \sum_{i=0}^{d} \sum_{j=0}^{d-i} b_{ij} x^i y^j$, one can compute their intersection points using:

- Resultants ($\mathrm{res}_y(A,B)$ and $\mathrm{res}_x(A,B)$),
- Gröbner bases [Buchberger 1965],
- Implicit representation of the y-coordinates.

Intersection of Ellipses

Consider the two ellipses:

$$\begin{aligned} A(x,y) &: \quad 4x^2 + 9y^2 - 36 = 0 \\ B(x,y) &: \quad 58x^2 + 72y^2 - 48xy - 360 = 0 \end{aligned}$$

The x-coordinates of the intersection points are roots of the polynomial:

$$\begin{aligned} \mathrm{res}_y(A,B) &= \det \begin{pmatrix} 9 & 0 & 4x^2-36 & 0 \\ 0 & 9 & 0 & 4x^2-36 \\ 72 & -48x & 58x^2-360 & 0 \\ 0 & 72 & -48x & 58x^2-360 \end{pmatrix} = \\ &= 137700x^4 - 1049760x^2 + 419904 \end{aligned}$$

$$\begin{aligned} x_1, \ldots, x_4 &= \{\pm 2.68328, \pm 0.65079\} \\ y_1, \ldots, y_4 &= \{\pm 1.95237, \pm 0.89443\} \end{aligned}$$

Algebraic Number Types (advanced)

Support of polynomial roots in the algebraic number libraries:

- CORE : `Expr`
  Supports the root-of operator (from Version 1.6) — only for polynomials with integer coefficients.
  [Li, Yap 2001]
- LEDA : `real`
  supports the diamond operator (pre-release) — diamond operations can be nested.
  [Burnikel, Funke, Mehlhorn, Schirra, Schmitt 2001]

Algebraic Curves and Surfaces

Robust predicates and constructions for arrangements of curves and surfaces:

- Conic curves and conic arcs
  [W 2002] [Berberich *et al.* 2002] [Emiris *et al.* 2004],
- Cubic curves
  [Eigenwillig, Kettner, Schömer, Wolpert 2004],
- Quartic curves
  [Berberich, Hemmer, Kettner, Schömer, Wolpert 2005],
- Algebraic curves
  [Seidel, Wolpert 2005],
- Quadric surfaces
  [Lazard, Peñaranda, Petitjean 2004] [Berberich *et al.* 2005].

One Step Beyond ...

- An exact Descartes algorithm with approximate coefficients [Eigenwillig, Kettner, Krandick, Mehlhorn, Schmitt, Wolpert 2005]:
  - Nested diamond operations,
  - Exact computation with rotations.
- Working with transcendental numbers [Yap 2006] — decidability of the zero problem (given $\alpha \in \mathbb{R}$, is $\alpha = 0$) in some cases:
  - Exact computations with arc lengths,
  - Exact computations with trigonometric functions,
  - Exact computations with logarithms and exponentials.